This Page Is Inserted by IFW Operations
and is not a part of the Official Record

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images may include (but are not limited to):

- BLACK BORDERS
- TEXT CUT OFF AT TOP, BOTTOM OR SIDES
- FADED TEXT
- ILLEGIBLE TEXT
- SKEWED/SLANTED IMAGES
- COLORED PHOTOS
- BLACK OR VERY BLACK AND WHITE DARK PHOTOS
- GRAY SCALE DOCUMENTS

IMAGES ARE BEST AVAILABLE COPY.

As rescanning documents *will not* correct images, please do not report the images to the Image Problems Mailbox.

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
8 March 2001 (08.03.2001)

PCT

(10) International Publication Number
WO 01/16742 A2

(51) International Patent Classification[7]: G06F 9/52

(21) International Application Number: PCT/US00/24248

(22) International Filing Date: 31 August 2000 (31.08.2000)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:

60/152,151	31 August 1999 (31.08.1999)	US
60/220,794	26 July 2000 (26.07.2000)	US
60/220,748	26 July 2000 (26.07.2000)	US

(63) Related by continuation (CON) or continuation-in-part (CIP) to earlier applications:

US	60/152,151 (CIP)
Filed on	31 August 1999 (31.08.1999)
US	60/220,794 (CIP)
Filed on	26 July 2000 (26.07.2000)
US	60/220,748 (CIP)
Filed on	26 July 2000 (26.07.2000)

(71) Applicant *(for all designated States except US)*: TIMES N SYSTEMS, INC. [US/US]; Building B, Suite P, 1908 Kramer Lane, Austin, TX 78758 (US).

(72) Inventor; and

(75) Inventor/Applicant *(for US only)*: MILLER, Chris [US/US]; 4807 Fieldstone Drive, Austin, TX 78735 (US).

(74) Agent: BRUCKNER, John, J.; Fulbright & Jaworski, L.L.P., Suite 2400, 600 Congress Avenue, Austin, TX 78701 (US).

(81) Designated States *(national)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CR, CU, CZ, DE, DK, DM, DZ, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, US, UZ, VN, YU, ZA, ZW.

(84) Designated States *(regional)*: ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Published:

— *Without international search report and to be republished upon receipt of that report.*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

WO 01/16742 A2

(54) Title: NETWORK SHARED MEMORY

(57) Abstract: Methods, systems and devices are described for a network shared memory. A method, includes: creating and assembling a message buffer in shared memory; parsing the message buffer for a plurality of pointer fields; and passing a message from a transmitting process to a receiving process by passing at least one pointer. The methods, systems and devices provide advantages because the speed and scalability of parallel processor systems is enhanced.

NETWORK SHARED MEMORY

BACKGROUND OF THE INVENTION

1. Field of the Invention

The invention relates generally to the field of computing systems in which multiple processors share some memory but in which each is required to provide separate access to standard network I/O implementation for interconnection. More particularly, the invention relates to computer science techniques that utilize a network shared memory.

2. Discussion of the Related Art

The clustering of workstations is a well-known art. In the most common cases, the clustering involves workstations that operate almost totally independently, utilizing the network only to share such services as a printer, license-limited applications, or shared files.

In more-closely-coupled environments, some software packages (such as NQS) allow a cluster of workstations to share work. In such cases the work arrives, typically as batch jobs, at an entry point to the cluster where it is queued and dispatched to the workstations on the basis of load.

In both of these cases, and all other known cases of clustering, the operating system and cluster subsystem are built around the concept of message-passing. The term message-passing means that a given workstation operates on some portion of a job until communications (to send or receive data, typically) with another workstation is necessary. Then, the first workstation prepares and communicates with the other workstation.

Another well-known art is that of clustering processors within a machine, usually called a Massively Parallel Processor or MPP, in which the techniques are essentially identical to those of clustered workstations. Usually, the bandwidth and latency of the interconnect network of an MPP are more highly optimized, but the system operation is the same.

In the general case, the passing of a message is an extremely expensive operation; expensive in the sense that many CPU cycles in the sender and

receiver are consumed by the process of sending, receiving, bracketing, verifying, and routing the message, CPU cycles that are therefore not available for other operations. A highly streamlined message-passing subsystem can typically require 10,000 to 20,000 CPU cycles or more.

There are specific cases wherein the passing of a message requires significantly less overhead. However, none of these specific cases is adaptable to a general-purpose computer system.

Message-passing parallel processor systems have been offered commercially for years but have failed to capture significant market share because of poor performance and difficulty of programming for typical parallel applications. Message-passing parallel processor systems do have some advantages. In particular, because they share no resources, message-passing parallel processor systems are easier to provide with high-availability features. What is needed is a better approach to parallel processor systems.

There are alternatives to the passing of messages for closely-coupled cluster work. One such alternative is the use of shared memory for inter-processor communication.

Shared-memory systems, have been much more successful at capturing market share than message-passing systems because of the dramatically superior performance of shared-memory systems, up to about four-processor systems. In Search of Clusters, Gregory F. Pfister 2nd ed. (January 1998) Prentice Hall Computer Books, ISBN: 0138997098 describes a computing system with multiple processing nodes in which each processing node is provided with private, local memory and also has access to a range of memory which is shared with other processing nodes. The disclosure of this publication in its entirety is hereby expressly incorporated herein by reference for the purpose of indicating the background of the invention and illustrating the state of the art.

However, providing high availability for traditional shared-memory systems has proved to be an elusive goal. The nature of these systems, which share all code and all data, including that data which controls the shared operating systems, is incompatible with the separation normally required for

high availability. What is needed is an approach to shared-memory systems that improves availability.

Although the use of shared memory for inter-processor communication is a well-known art, prior to the teachings of U.S. Ser. No. 09/273,430, filed March 19, 1999, entitled Shared Memory Apparatus and Method for Multiprocessing Systems, the processors shared a single copy of the operating system. The problem with such systems is that they cannot be efficiently scaled beyond four to eight way systems except in unusual circumstances. All known cases of said unusual circumstances are such that the systems are not good price-performance systems for general-purpose computing.

The entire contents of U.S. Patent Applications 09/273,430, filed March 19, 1999 and PCT/US00/01262, filed January 18, 2000 are hereby expressly incorporated by reference herein for all purposes. U.S. Ser. No. 09/273,430, improved upon the concept of shared memory by teaching the concept which will herein be referred to as a tight cluster. The concept of a tight cluster is that of individual computers, each with its own CPU(s), memory, I/O, and operating system, but for which collection of computers there is a portion of memory which is shared by all the computers and via which they can exchange information. U.S. Ser. No. 09/273,430 describes a system in which each processing node is provided with its own private copy of an operating system and in which the connection to shared memory is via a standard bus. The advantage of a tight cluster in comparison to an SMP is "scalability" which means that a much larger number of computers can be attached together via a tight cluster than an SMP with little loss of processing efficiency.

What is needed are improvements to the concept of the tight cluster. What is also needed is an expansion of the concept of the tight cluster.

SUMMARY OF THE INVENTION

A goal of the invention is to simultaneously satisfy the above-discussed requirements of improving and expanding the tight cluster concept which, in the case of the prior art, are not satisfied.

One embodiment of the invention is based on a method, comprising: creating and assembling a message buffer in shared memory; parsing said message buffer for a plurality of pointer fields; and passing a message from a transmitting process to a receiving process by passing at least one pointer. Another embodiment of the invention is based on an apparatus, comprising: a shared memory node; a first processing node coupled to said shared memory node; and a second processing node coupled to said shared memory node, wherein a message buffer is created and assembled in said shared memory node, said message buffer is parsed for a plurality of pointer fields; and a message from said first processing node to said second processing node is passed by at least one pointer. Another embodiment of the invention is based on an electronic media, comprising: a computer program adapted to create and assemble a message buffer in shared memory; parse said message buffer for a plurality of pointer fields; and pass a message from a transmitting process to a receiving process by passing at least one pointer. Another embodiment of the invention is based on a computer program comprising computer program means adapted to perform the steps of: creating and assembling a message buffer in shared memory; parsing said message buffer for a plurality of pointer fields; and passing a message from a transmitting process to a receiving process by passing at least one pointer when said computer program is run on a computer. Another embodiment of the invention is based on a system, comprising a multiplicity of processors, each with some private memory and the multiplicity with some shared memory, interconnected and arranged such that memory accesses to a first set of address ranges will be to local, private memory whereas memory accesses to a second set of address ranges will be to shared memory, and configured so that MBUFs are constructed and connected within shared memory. Another embodiment of the invention is based on a computer system comprising Operating System extensions to perform network I/O functions in a shared-memory environment, wherein said Operating System extensions perform the functions with Load and Store operations. Another embodiment of the invention is based on a computer system comprising Operating System extensions to perform network I/O functions in a shared-memory environment,

wherein said Operating System extensions transparently simulate standard networking protocols. Another embodiment of the invention is based on an apparatus, comprising: a shared memory node; a first processing node coupled to said shared memory node; and a second processing node coupled to said shared memory node, wherein Operating System extensions perform network I/O functions in a shared-memory environment with Load and Store operations. Another embodiment of the invention is based on an apparatus, comprising: a shared memory node; a first processing node coupled to said shared memory node; and a second processing node coupled to said shared memory node, wherein Operating System extensions perform network I/O functions in a shared-memory environment and transparently simulate standard networking protocols.

These, and other goals and embodiments of the invention will be better appreciated and understood when considered in conjunction with the following description and the accompanying drawing. It should be understood, however, that the following description, while indicating preferred embodiments of the invention and numerous specific details thereof, is given by way of illustration and not of limitation. Many changes and modifications may be made within the scope of the invention without departing from the spirit thereof, and the invention includes all such modifications.

BRIEF DESCRIPTION OF THE DRAWING

A clear conception of the advantages and features constituting the invention, and of the components and operation of model systems provided with the invention, will become more readily apparent by referring to the exemplary, and therefore nonlimiting, embodiments illustrated in the drawing accompanying and forming a part of this specification, wherein like reference characters (if they occur in more than one view) designate the same parts. It should be noted that the features illustrated in the drawing are not necessarily drawn to scale.

FIG. 1 illustrates a block schematic view of a system, representing an embodiment of the invention.

DESCRIPTION OF PREFERRED EMBODIMENTS

The invention and the various features and advantageous details thereof are explained more fully with reference to the nonlimiting embodiments that are illustrated in the accompanying drawing and detailed in the following description of preferred embodiments. Descriptions of well known components and processing techniques are omitted so as not to unnecessarily obscure the invention in detail.

The teachings of U.S. Ser. No. 09/273,430 include a system which is a single entity; one large supercomputer. The invention is also applicable to a cluster of workstations, or even a network.

The invention is applicable to systems of the Pfister or the type of U.S. Ser. No. 09/273,430 in which each processing node has its own copy of an operating system. The invention is also applicable to other types of multiple processing node systems.

The context of the invention can include a tight cluster as described in U.S. Ser. No. 09/273,430. A tight cluster is defined as a cluster of workstations or an arrangement within a single, multiple-processor machine in which the processors are connected by a high-speed, low-latency interconnection, and in which some but not all memory is shared among the processors. Within the scope of a given processor, accesses to a first set of ranges of memory addresses will be to local, private memory but accesses to a second set of memory address ranges will be to shared memory. The significant advantage to a tight cluster in comparison to a message-passing cluster is that, assuming the environment has been appropriately established, the exchange of information involves a single STORE instruction by the sending processor and a subsequent single LOAD instruction by the receiving processor.

The establishment of the environment, taught by U.S. Ser. No. 09/273,430 and more fully by companion disclosures (U.S. Provisional Application Ser. No. 60/220,794, filed July 26, 2000; U.S. Provisional Application Ser. No. 60/220,748, filed July 26, 2000; WSGR 15245-711; WSGR 15245-712; WSGR 15245-713; WSGR 15245-715; WSGR 15245-716;

WSGR 15245-717; WSGR 15245-718; and WSGR 15245-720, the entire contents of all which are hereby expressly incorporated herein by reference for all purposes) can be performed in such a way as to require relatively little system overhead, and to be done once for many, many information exchanges. Therefore, a comparison of 10,000 instructions for message-passing to a pair of instructions for tight-clustering, is valid.

In a shared-memory cluster, some memory is private to each processor and some memory is shared among the processors. This invention can include utilizing the shared memory for exchanging information among the processors. The information can be passed using existing network interfaces through shared memory. For a computing system in which multiple processing nodes share some memory, and where each node requires standard networking methods of interconnect, the invention can include the utilization of shared memory to achieve OS-transparent high-speed access to network resources used for interconnecting said nodes.

Within an MPP or a cluster of workstations, or even a network of more-loosely coupled computers are generally-used protocols for the sending of messages. These protocols assure robust delivery of messages over widely-dispersed, often flimsy networks. The protocols used therein are therefore quite ill-suited to simple, secure, intercommunication media, particularly to media with secure memory.

However, these protocols are so widespread that a vast array of applications and subsystems utilize them. The result is that most applications and subsystems provide well-defined interfaces, interfaces that have been developed to meet with these particular protocols.

These interfaces produce complex, slow transfer of data. Two basic kinds of complexities are introduced by these packages which result in reliable communications over unreliable, complex networks. However, these complexities cause complex, slow communications within shared memory.

The first of these complexities is one that will here be called layering. When a first node in a complex network develops a message to pass to a second node within the network, the standard packages generally require that a massive

amount of "layering" of the communication subsystem occur. The message is transformed, step-by-step, from a simple buffer containing application-level data into a message suitable for network traffic. Each step is performed within a given layer of the communication subsystem, and these layers pass little information from the one to the next. The first layer provides a first transformation to the data and passes the transformed information to the second which transforms that information and passes the result to the third layer, and in similar fashion down through the layers. No layer passes information to the next describing anything about the incoming information nor about the transformation performed except for the basic information necessary to continue the process of preparation for transmission over a complex network. Therefore, only with difficulty can the final result be examined by an automated process to determine the original information submitted.

The second, companion complexity will here be called MBUFs. Not only does each layer transform and repackage the information from the layer above, but also each of several of the various layers break the information each respectively receives from the layer above into finer and finer entities (MBUFs, or message buffers). For an automated process to locate and correlate the final MBUFs to the original application data is difficult.

U.S. Ser. No. 09/273,430 describes a computing system in which multiple processing nodes are provided; and in which each is provided with some local, private memory, and further in which all have access to a portion of memory which is shared. U.S. Ser. No. 09/273,430 teaches the sharing of memory via a standard bus. In U.S. Ser. No. 09/273,430 each processing node is provided with separate networking I/O implemented over standard serial media.

The invention can be used with the kind of systems taught by U.S. Ser. No. 09/273,430. The invention is also applicable to other architectures such as NUMA machines in which each processor or processor aggregation is provided with a separate network I/O facility.

The invention can be used in an environment as described in U.S. Ser. No. 09/273,430 where multiple computers selectively address a first set of memory address ranges which will be to private memory and a second set of

processed. The network I/O functions can be processed by the NSM extensions and be translated into shared-memory Load and Store instructions. In this way, the NSM extensions satisfy the Operating System I/O request transparently.

The NSM extensions can simulate the behavior of a standard networking media, such as Ethernet, Token Ring, etceteras. Packet sizes appropriate to the medium can be supported transparently, allowing, for example, the large packet sizes of Token Ring to be exploited to minimize network protocol stack message fragmentation. The standardized behavior of Ethernet can be used for those application implementations requiring it.

A more specific preferred implementation will now be described. When a NSM node requires a standard networking message to be sent to another node, the NSM extensions will copy the message out to shared-memory. The medium-appropriate destination address is then used as input into a hash function, which results in a table lookup returning the target node's address, as per the requirements of the primitives. To one skilled in the art, the network address to target address lookup is a straightforward implementation. The target node is then notified of the packet presence and its address in shared memory through mechanisms provided by Load and Store instructions. The target node may then indicate the data to the Operating System as appropriate, copying the shared-memory data as required.

If the target address is not in the lookup table, the sending node can use mechanisms provided by Load and Store instructions to broadcast to all nodes, ensuring that every node gets notified that the packet is present in shared memory. As each node examines the packet header, standard networking implementations dictate that it will ignore a message whose destination address it does not recognize.

When the receiving node issues a response to the network message, it will likewise be sent to all nodes via the broadcast mechanism previously described. Upon reception of notification of data in shared memory that matches the simulated destination address, the previous sending node (now the receiving node), can place the address in the hash table for lookup the next time a send is required.

The hash table entries may be aged; discarding them after time. The fixed configuration nature of a group shared-memory connected to processing nodes, however, does not require this as a standard networking interconnect implementation would. The operating system perceived implementation of the networking interconnect can remain, in all cases, transparent.

While not being limited to any particular performance indicator or diagnostic identifier, preferred embodiments of the invention can be identified one at a time by testing for the substantially highest performance. The test for the substantially highest performance can be carried out without undue experimentation by the use of a simple and conventional benchmark (speed) experiment.

The term substantially, as used herein, is defined as at least approaching a given state (e.g., preferably within 10% of, more preferably within 1% of, and most preferably within 0.1% of). The term coupled, as used herein, is defined as connected, although not necessarily directly, and not necessarily mechanically. The term means, as used herein, is defined as hardware, firmware and/or software for achieving a result. The term program or phrase computer program, as used herein, is defined as a sequence of instructions designed for execution on a computer system. A program may include a subroutine, a function, a procedure, an object method, an object implementation, an executable application, an applet, a servlet, a source code, an object code, and/or other sequence of instructions designed for execution on a computer system.

Practical Applications of the Invention

A practical application of the invention that has value within the technological arts is waveform transformation. Further, the invention is useful in conjunction with data input and transformation (such as are used for the purpose of speech recognition), or in conjunction with transforming the appearance of a display (such as are used for the purpose of video games), or the like. There are virtually innumerable uses for the invention, all of which need not be detailed here.

Advantages of the Invention

A system, representing an embodiment of the invention, can be cost effective and advantageous for at least the following reasons. The invention improves the speed of parallel computing systems. The invention improves the scalability of parallel computing systems.

All the disclosed embodiments of the invention described herein can be realized and practiced without undue experimentation. Although the best mode of carrying out the invention contemplated by the inventors is disclosed above, practice of the invention is not limited thereto. Accordingly, it will be appreciated by those skilled in the art that the invention may be practiced otherwise than as specifically described herein.

For example, although the network shared memory described herein can be a separate module, it will be manifest that the network shared memory may be integrated into the system with which it is associated. Furthermore, all the disclosed elements and features of each disclosed embodiment can be combined with, or substituted for, the disclosed elements and features of every other disclosed embodiment except where such elements or features are mutually exclusive.

It will be manifest that various additions, modifications and rearrangements of the features of the invention may be made without deviating from the spirit and scope of the underlying inventive concept. It is intended that the scope of the invention as defined by the appended claims and their equivalents cover all such additions, modifications, and rearrangements.

The appended claims are not to be interpreted as including means-plus-function limitations, unless such a limitation is explicitly recited in a given claim using the phrase "means for." Expedient embodiments of the invention are differentiated by the appended subclaims.

CLAIMS

What is claimed is:

1. A method, comprising:

creating and assembling a message buffer in shared memory;

parsing said message buffer for a plurality of pointer fields; and

passing a message from a transmitting process to a receiving process by passing at least one pointer.

2. The method of claim 1, wherein the pointer points to a head of an MBUF chain and the receiving process can read a plurality of message buffers by merely following successive message buffer pointers.

3. The method of claim 1, further comprising processing network I/O functions with network-shared-memory extensions to translate network I/O functions into shared-memory Load and Store instructions.

4. An apparatus, comprising:

a shared memory node;

a first processing node coupled to said shared memory node; and

a second processing node coupled to said shared memory node,

wherein a message buffer is created and assembled in said shared memory node, said message buffer is parsed for a plurality of pointer fields; and a message from said first processing node to said second processing node is passed by at least one pointer.

5. A computer system, comprising the apparatus of claim 4.

6. An electronic media, comprising: a computer program adapted to create and assemble a message buffer in shared memory; parse said message buffer for a plurality of pointer fields; and pass a message from a transmitting process to a receiving process by passing at least one pointer.

7. A computer program comprising computer program means adapted to perform the steps of

creating and assembling a message buffer in shared memory;

parsing said message buffer for a plurality of pointer fields; and

passing a message from a transmitting process to a receiving process by passing at least one pointer when said computer program is run on a computer.

8. A computer program as claimed in claim 7, embodied on a computer-readable medium.

9. A system, comprising a multiplicity of processors, each with some private memory and the multiplicity with some shared memory, interconnected and arranged such that memory accesses to a first set of address ranges will be to local, private memory whereas memory accesses to a second set of address ranges will be to shared memory, and configured so that MBUFs are constructed and connected within shared memory.

10. The system of claim 9, wherein the system is configured so that the pointers from each buffer to the next are identified, and the offset within each to user data is specified.

11. The system of claim 10, wherein passing of the message includes passing a pointer to the first MBUF.

12. The system of claim 9, wherein each processor is provided with a pool of outgoing MBUFs within shared memory and uses MBUFs from that pool to send messages to other processors.

13. The system of claim 9, wherein all processors are provided with a common pool of outgoing MBUFs within shared memory and use MBUFs from that pool to send messages to other processors.

14. A computer system comprising Operating System extensions to perform network I/O functions in a shared-memory environment, wherein said Operating System extensions perform the functions with Load and Store operations.

15. A computer system comprising Operating System extensions to perform network I/O functions in a shared-memory environment, wherein said Operating System extensions transparently simulate standard networking protocols.

16. An apparatus, comprising:
a shared memory node;
a first processing node coupled to said shared memory node; and
a second processing node coupled to said shared memory node,
wherein Operating System extensions perform network I/O functions in a shared-memory environment with Load and Store operations.

17. An apparatus, comprising:
a shared memory node;
a first processing node coupled to said shared memory node; and
a second processing node coupled to said shared memory node,
wherein Operating System extensions perform network I/O functions in a shared-memory environment and transparently simulate standard networking protocols.

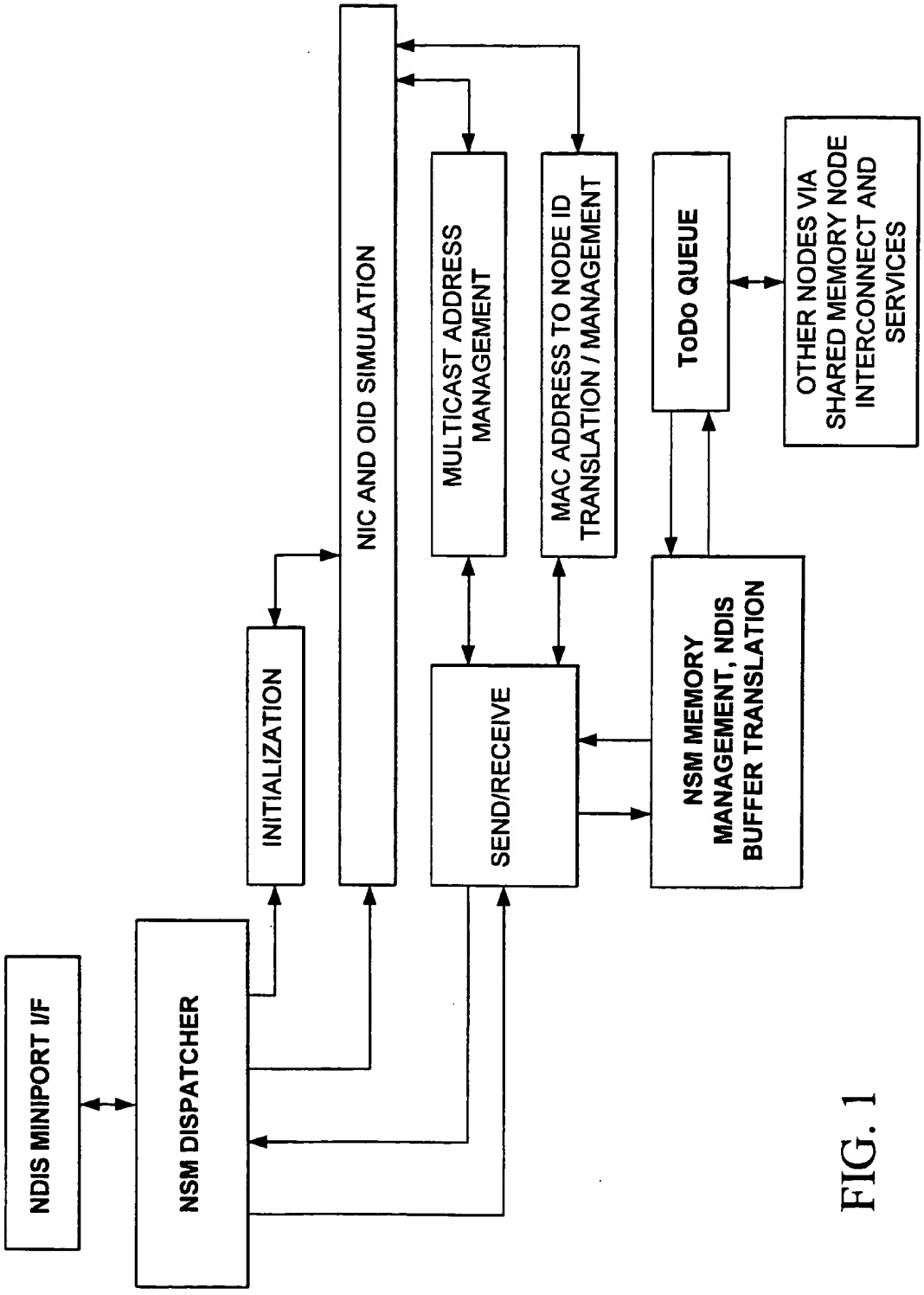
NDIS MINIPORT I/F
NSM DISPATCHER
INITIALIZATION
NIC AND OID SIMULATION
SEND/RECEIVE
MULTICAST ADDRESS MANAGEMENT
MAC ADDRESS TO NODE ID TRANSLATION / MANAGEMENT
NSM MEMORY MANAGEMENT, NDIS BUFFER TRANSLATION
ToDo QUEUE
OTHER NODES VIA SHARED MEMORY NODE INTERCONNECT AND SERVICES

FIG. 1

memory ranges which will be to shared memory. The invention can also be used in an environment that includes a large number of existing packages for the interchange of data and which meet the interfaces required for using those packages. The invention can include emulating those packages while simultaneously achieving highly efficient, fast, reliable transfer of information within a shared-memory cluster, and while avoiding the need to completely rewrite the protocols.

The first step of the invention is to redirect a MBUF subsystem so that the MBUFs are created and assembled in shared memory. This is in contrast to assembling the MBUFs in the private memory of the processing node.

The second step is to parse each MBUF for the pointer fields and to pull out these pointers separately. The passing of a message can, therefore, consist only of the passing of a pointer to the receiving process; the pointer points to the head of the MBUF chain, and the receiving process can read the buffers by merely following the successive MBUF pointers. No movement of data is necessary, and no message is therefore physically passed. In this manner, a message of many megabytes can be passed by the mere passing only of a one-word pointer.

The invention can be embodied outside the operating system but within shared memory to provide any node in a shared-memory computing system access to a shared memory, which is physically attached to another node. In a preferred embodiment, each processing node is provided with some local, private memory and a separate copy of the operating system. In this preferred embodiment, each of several nodes is provided with its own I/O channel to disks, networking adapters, and other I/O units. In this preferred embodiment, the operating system in each node is augmented with external extensions (not part of the operating system) which that can reach shared memory and communicate to other nodes via Load and Store instructions to shared memory.

Referring to FIG. 1, the invention can include other extensions, called Network-Shared-Memory (NSM) extensions, which make use of primitives of Load and Store instructions and by which network communication functions, which originate in applications and Operating System internal functions, can be